
Approximation Algorithms

Chapter 6: Review and Relay Placement

Sándor P. Fekete

Algorithms Division
Department of Computer Science
TU Braunschweig



- 1. Introduction**
- 2. Review**
- 3. Extra Packing: Dispersion**
- 4. Extra Tours: Lawn Mowing**
- 5. Relay Placement**
- 6. Coordinated Motion Planning**

1. Introduction
2. Review
3. Extra Packing: Dispersion
4. Extra Tours: Lawn Mowing
5. Relay Placement
6. Coordinated Motion Planning

Review: Matching & Vertex Cover

An example

Given: A graph $G=(V,E)$

Wanted: A minimum vertex cover (VC):
a set $S \subseteq V$ of smallest possible cardinality,
such that for every edge $e=(v_1,v_2) \in E$,

$$\{v_1, v_2\} \cap S \neq \emptyset$$

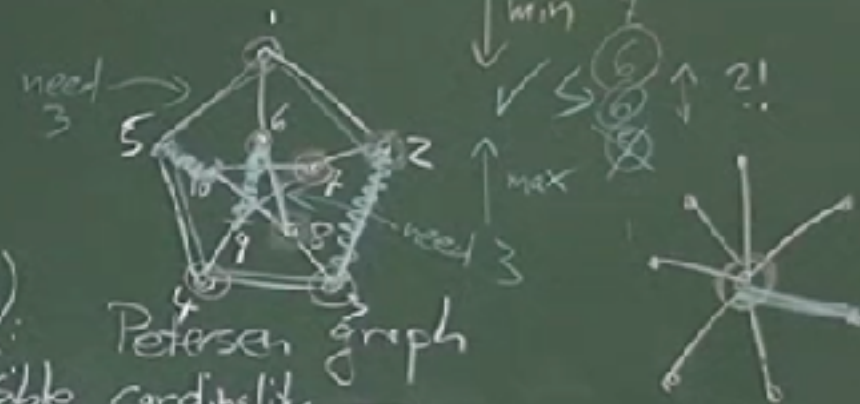
Wanted!

Given: Graph $G=(V,E)$

Wanted: A maximum matching:

a set $M \subseteq E$ of largest
cardinality, such that for every two
 $e_1, e_2 \in M$: $e_1 \cap e_2 = \emptyset$

Approximation Algorithms



Review: Approximation Algorithms

Given: An NP-complete (or NP-hard) problem Π

Wanted: An approximation algorithm for Π :

(1) Runtime is polynomial.

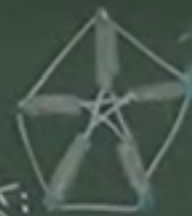
(2) There is a constant c , such that for any instance I of Π , the algorithm computes a solution of value within c of the optimum value.

For minimization: $\text{APPROX}(I) \leq c \cdot \text{OPT}(I) \rightarrow c \geq 1$

For maximization: $\text{APPROX}(I) \geq c \cdot \text{OPT}(I) \rightarrow c \leq 1$

Wanted: An AA for VC!

Idea: Consider a maximal matching. Pick both vertices for each edge.



To check:
(0)
(1)
(2)

Veget
a VC!
factor

Review: Set Cover

Set cover

Idea: Consider the relative cost (cost per element) for the currently uncovered elements

Approximation algorithm for SET COVER: Greedy

Edges

42 ← cost per element

Algorithm 2.2 (Greedy set cover)

1. $C \leftarrow \emptyset$ // C : covered elements
2. WHILE ($C \neq U$) DO
 - Find the most cost-efficient set in the current iteration, say S
 - Let $\alpha = \frac{\text{cost}(S)}{|S - C|}$, i.e., the cost-efficiency of S .

• Pick S , and for each $e \in S$, set price

Review: Maximum Coverage

Now: GREEDY is never worse than an H_n -approximation.

Important step:

Lemma 2.3

For each $k \in \{1, \dots, n\}$ in the sequence $e_1, \dots, e_n$ in which elements are covered by GREEDY, we have $\text{price}(e_k) \leq \frac{\text{OPT}}{n-k+1}$

THEOREM 2.4

The GREEDY algorithm is an H_n -approximation algorithm for SET COVER.

Proof:

The total cost for GREEDY is $\sum_{k=1}^n \text{cost}(e_k)$
 $\leq \sum_{k=1}^n \frac{\text{OPT}}{n-k+1} = \text{OPT} \cdot \sum_{k=1}^n \frac{1}{k} = \text{OPT} \cdot H_n$

Different but related:

PROBLEM 2.4½ (Maximum Coverage)

Int: Same as for SET COVER, plus a budget (e.g. a number k that tells us how many sets we can pick)

Want: Cover as many elements as possible



Review: Shortest Superstring

LEMMA 2.11 $OPT \leq OPT_{\phi} \leq 2OPT$

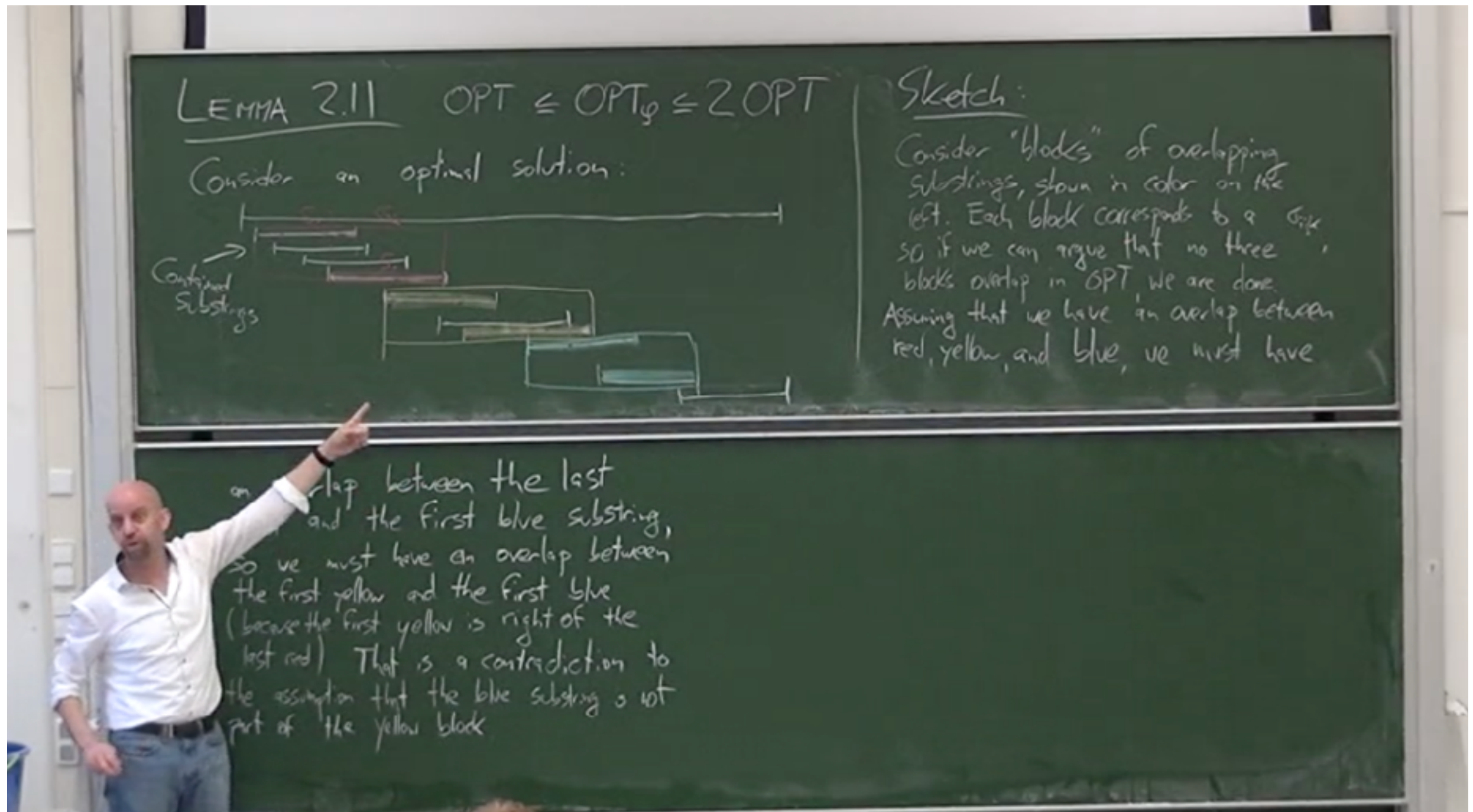
Consider an optimal solution:

Continued Substrings

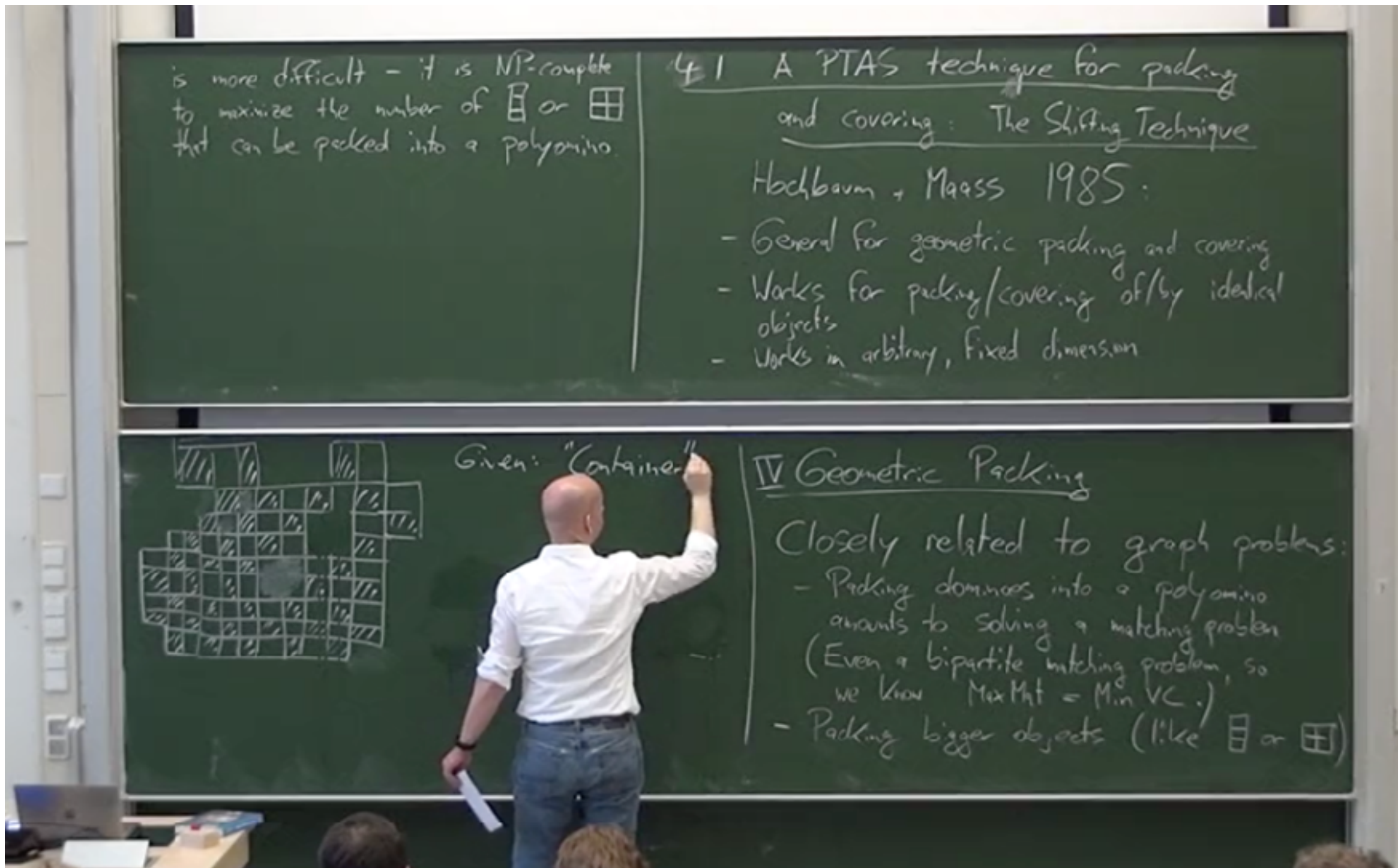
Sketch:

Consider "blocks" of overlapping substrings, shown in color on the left. Each block corresponds to a ϕ -size, so if we can argue that no three blocks overlap in OPT , we are done. Assuming that we have an overlap between red, yellow, and blue, we must have

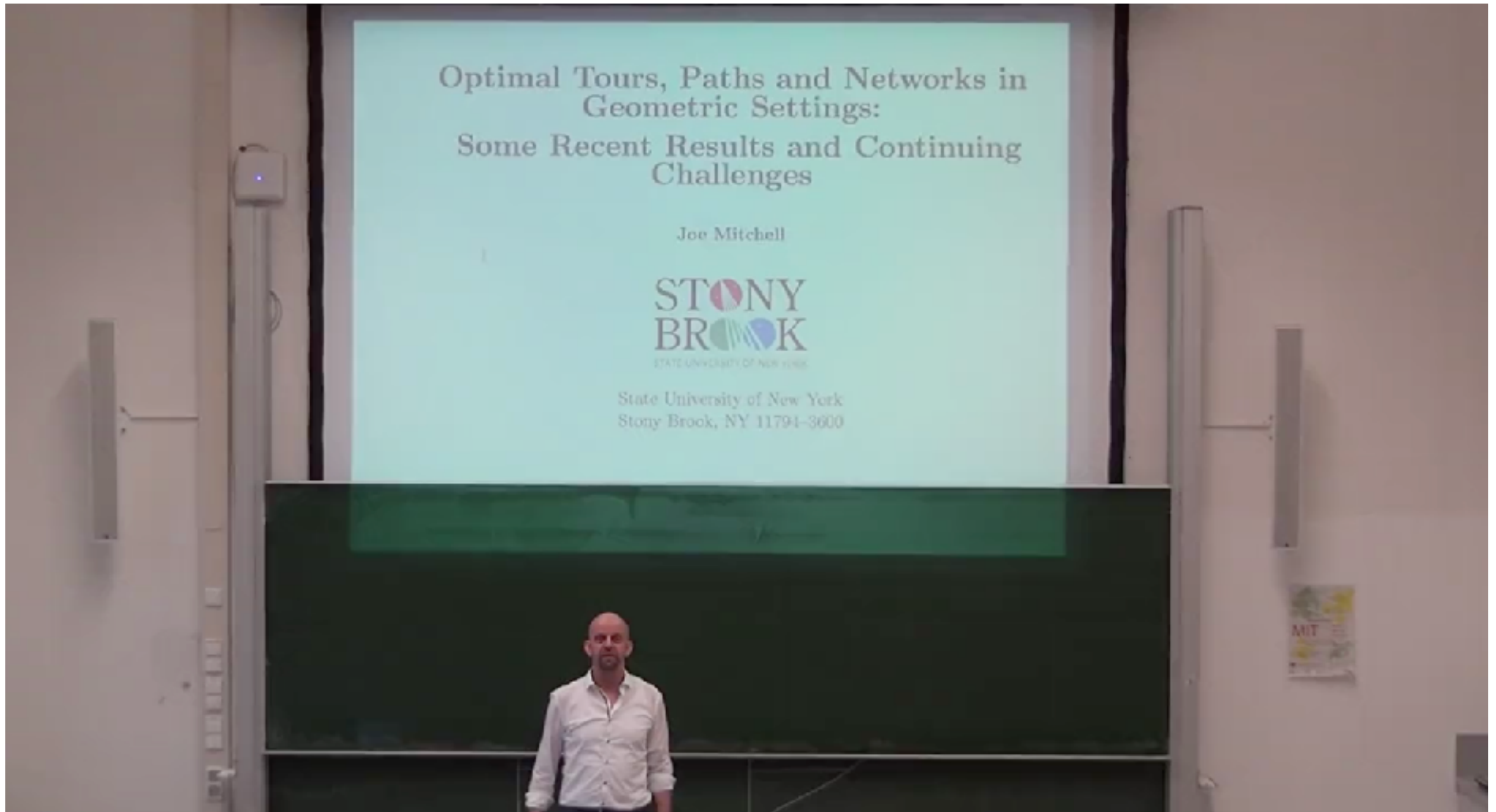
an overlap between the last and the first blue substring, so we must have an overlap between the first yellow and the first blue (because the first yellow is right of the last red). That is a contradiction to the assumption that the blue substring is not part of the yellow block.



Review: Packing



Review: Tour Problems



Review: Tour Problems



The Freeze-Tag Problem: How to Wake Up a Swarm of Robots

Estie Arkin¹

Michael Bender¹

Sándor Fekete²

Joe Mitchell¹

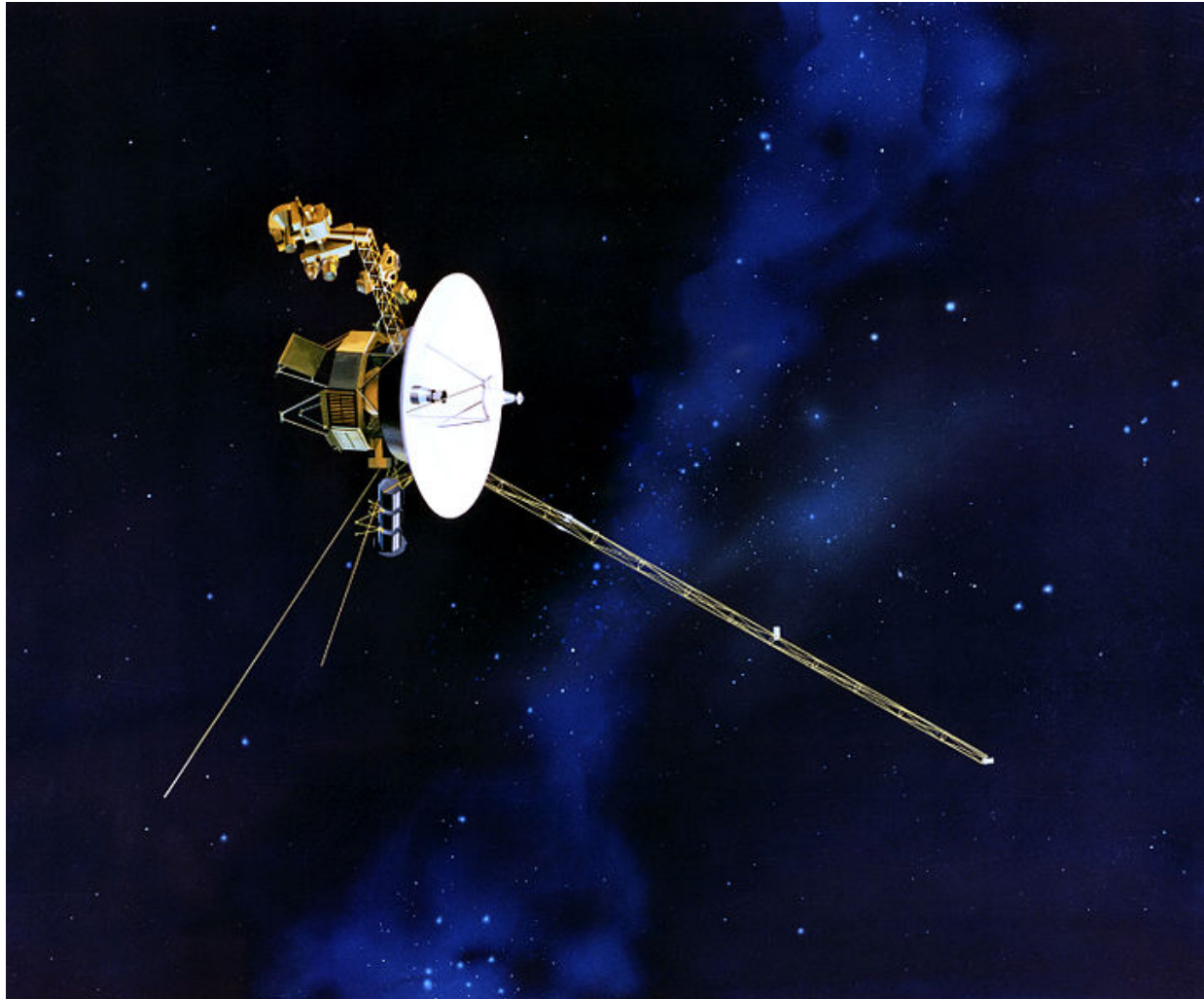
Martin Skutella³

¹ University at Stony Brook

² TU Braunschweig

³ TU Berlin

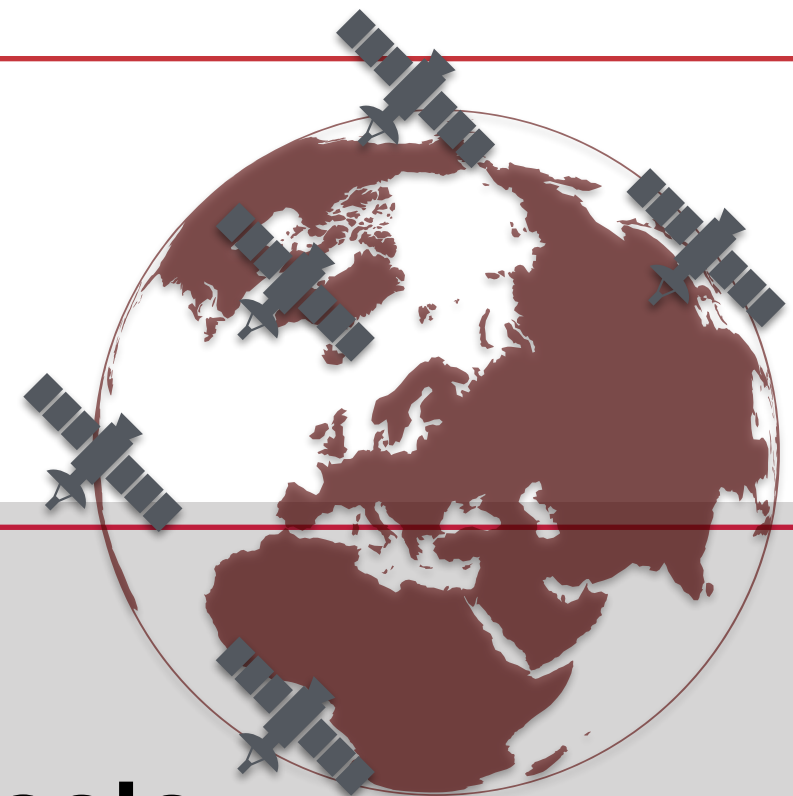
Motivation



- Highly focused antennas.
- Expensive rotations.

How can we quickly distribute information from one satellite to all others?

Minimum Scan Cover with Angular Transition Costs



Sándor Fekete



Linda Kleist



Dominik Krupke

1. Introduction
2. Review
3. **Extra Packing: Dispersion**
4. **Extra Tours: Lawn Mowing**
5. **Relay Placement**
6. **Coordinated Motion Planning**

Dispersion (1)

Algorithmica (2001) 30: 451–470
DOI: 10.1007/s00453-001-0022-x

Algorithmica
© 2001 Springer-Verlag New York Inc.

Approximation of Geometric Dispersion Problems¹

C. Baur² and S. P. Fekete³

Abstract. We consider problems of distributing a number of points within a polygonal region P , such that the points are “far away” from each other. Problems of this type have been considered before for the case where the possible locations form a discrete set. Dispersion problems are closely related to packing problems. While Hochbaum and Maass [20] have given a polynomial-time approximation scheme for packing, we show that geometric dispersion problems cannot be approximated arbitrarily well in polynomial time, unless $P = NP$. A special case of this observation solves an open problem by Rosenkrantz et al. [31]. We give a $\frac{2}{3}$ approximation algorithm for one version of the geometric dispersion problem. This algorithm is strongly polynomial in the size of the input, i.e., its running time does not depend on the area of P . We also discuss extensions and open problems.

Key Words. Packing, Dispersion, Location problems, Geometric optimization, Bounds on approximation factors.

Dispersion (2)

$\text{PACK}(k, L)$

Input: A polygonal region P with n vertices, a parameter k , a parameter L .

Question: Can k many L -squares be packed into P ?



$\max_k \text{PACK}(L)$

Input: A polygonal region P with n vertices.

Task: Pack k many L -squares into P , such that k is as big as possible.

$\max_L \text{PACK}(k)$

Input: A polygonal region P with n vertices.

Task: Pack k many $L \times L$ squares into P , such that L is as big as possible.

Dispersion (2)

$\text{PACK}(k, L)$

Input: A polygonal region P with n vertices, a parameter k , a parameter L .

Question: Can k many L -squares be packed into P ?



$\max_k \text{PACK}(L)$

Input: A polygonal region P with n vertices.

Task: Pack k many L -squares into P , such that k is as big as possible.

$\max_L \text{PACK}(k)$

Input: A polygonal region P with n vertices.

Task: Pack k many $L \times L$ squares into P , such that L is as big as possible.



Dispersion (3)

THEOREM 1. *Unless $P = NP$, there is no polynomial algorithm that finds a solution within more than $\frac{13}{14}$ of the optimum for rectilinear geometric dispersion with boundaries.*

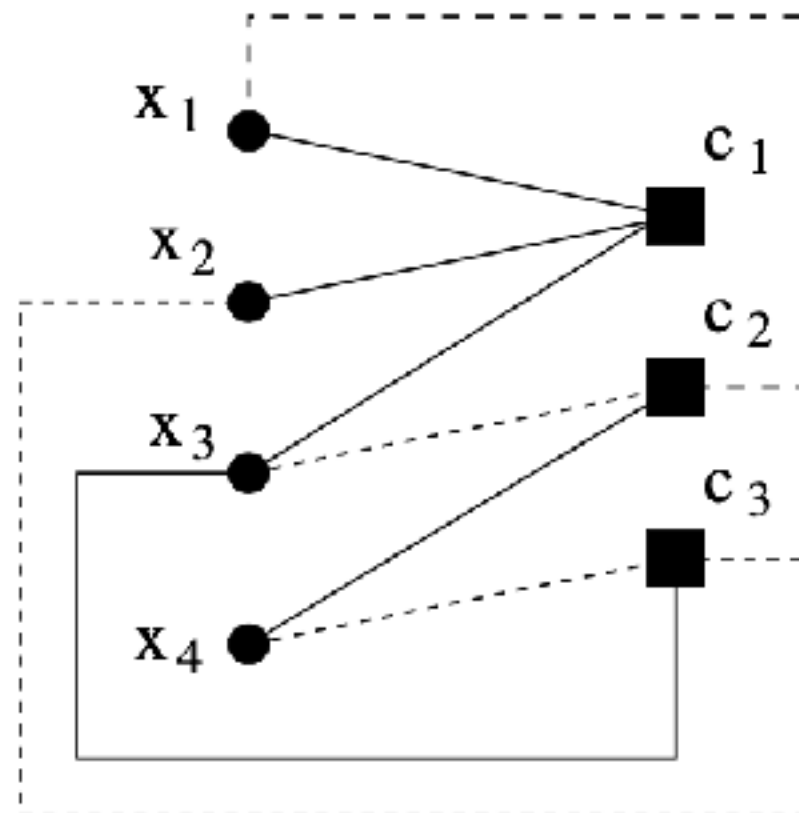


Fig. 1. The graph G_I representing the PLANAR 3SAT instance $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4)$. Edges are distinguished according to the logical parity of the corresponding literals.

Dispersion (3)

THEOREM 1. *Unless $P = NP$, there is no polynomial algorithm that finds a solution within more than $\frac{13}{14}$ of the optimum for rectilinear geometric dispersion with boundaries.*

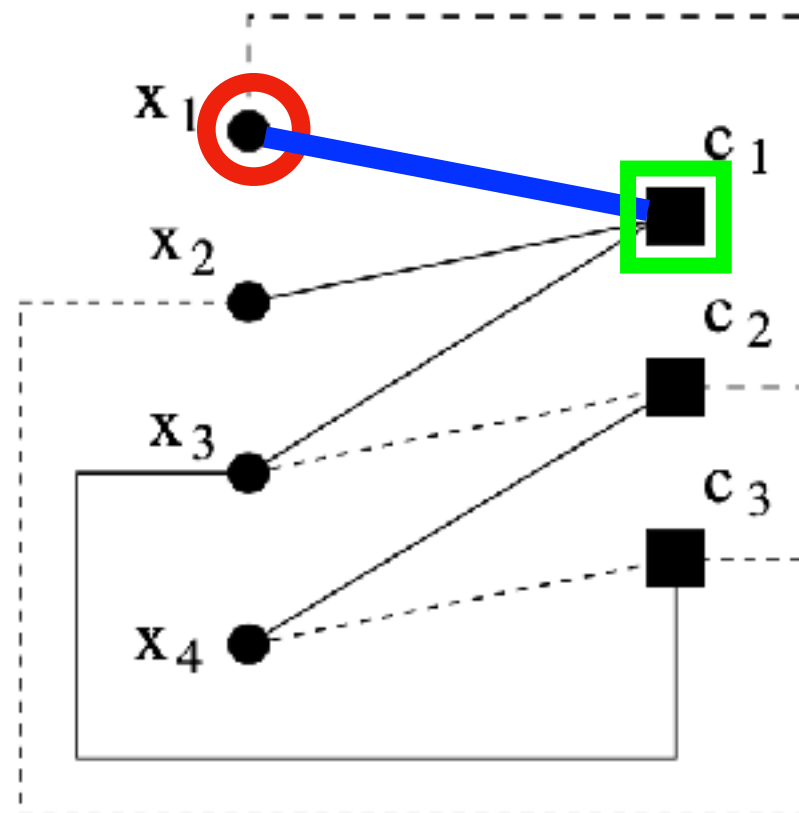
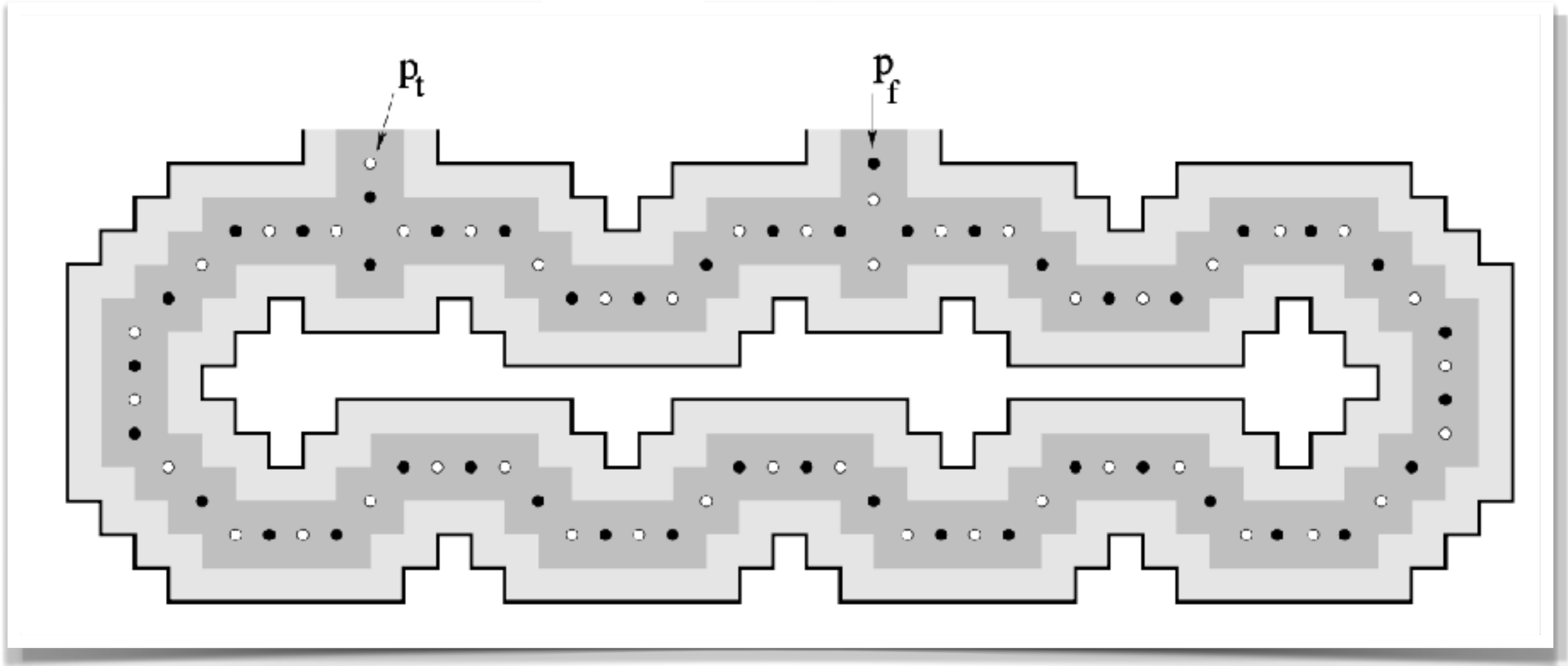
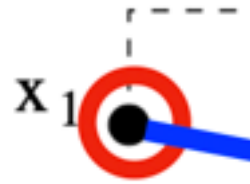
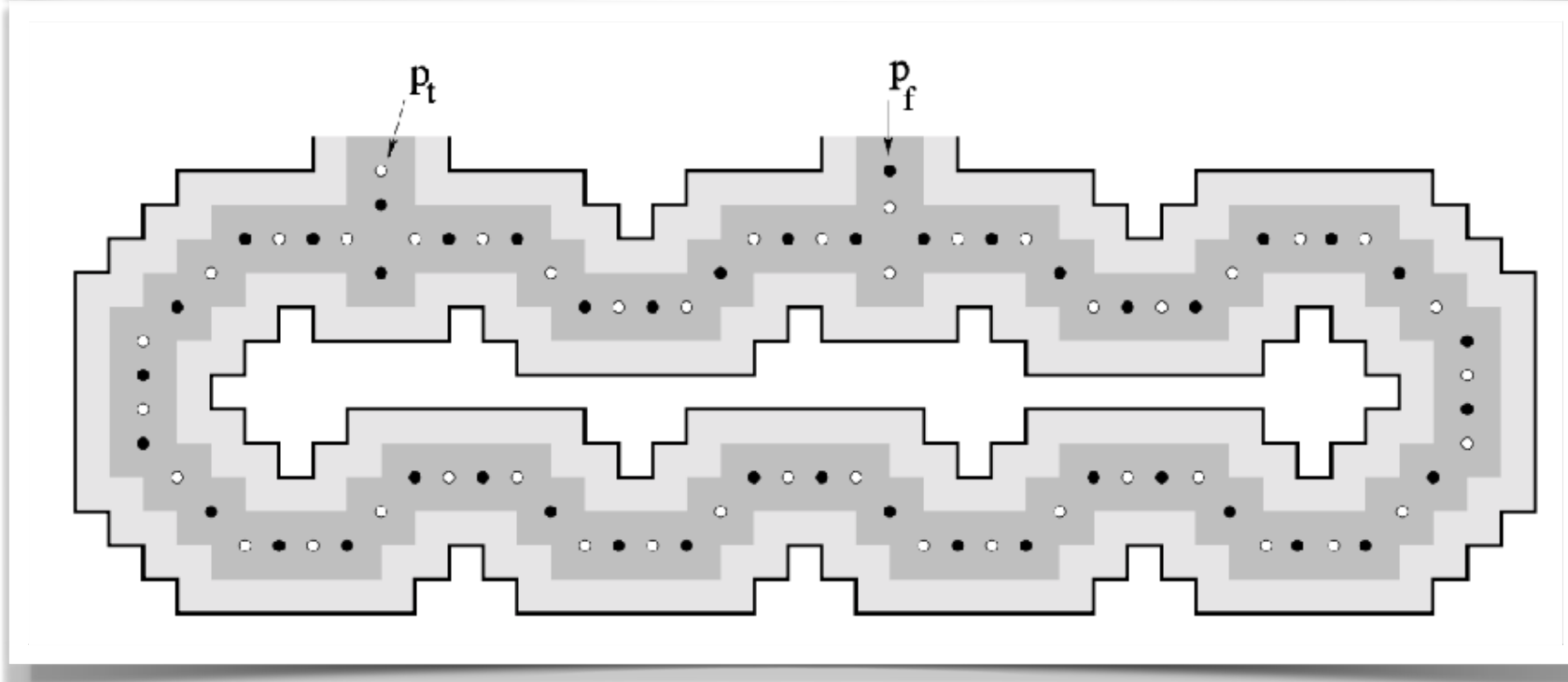
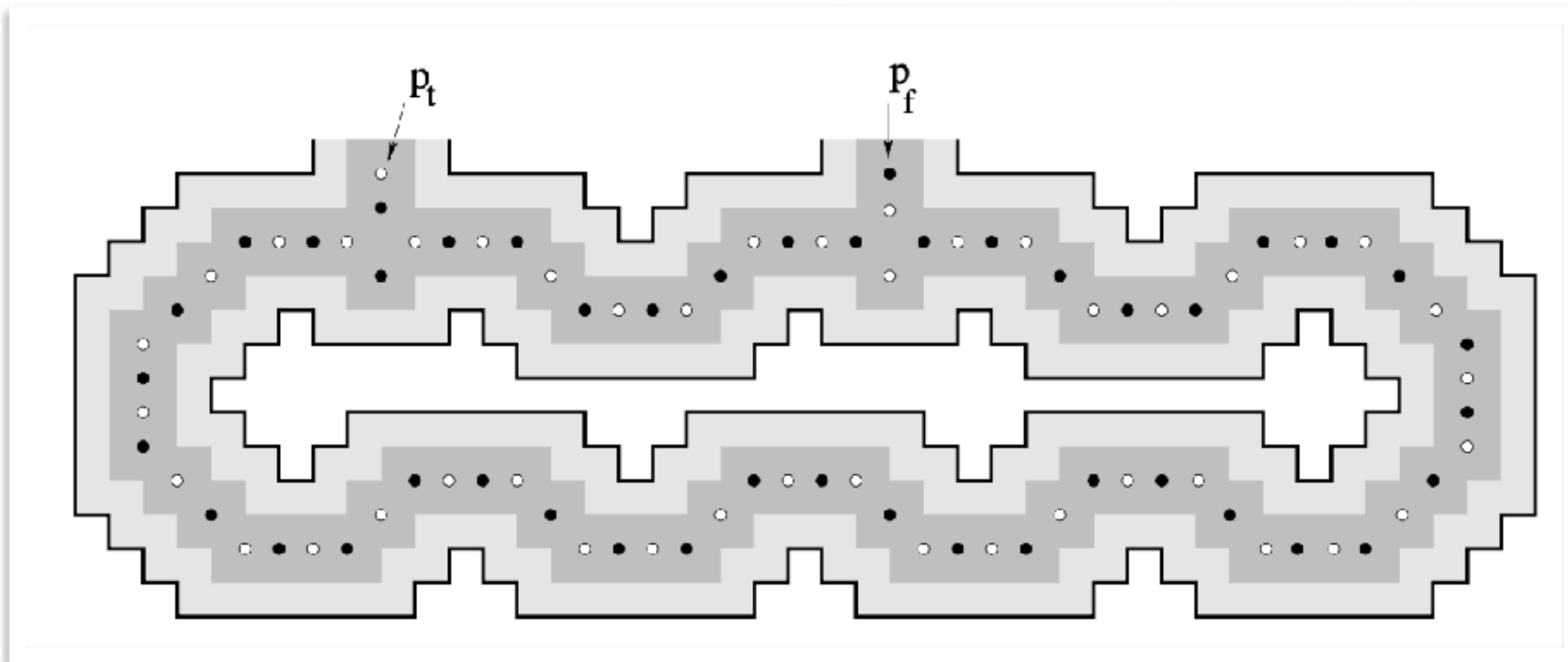


Fig. 1. The graph G_I representing the PLANAR 3SAT instance $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4)$. Edges are distinguished according to the logical parity of the corresponding literals.

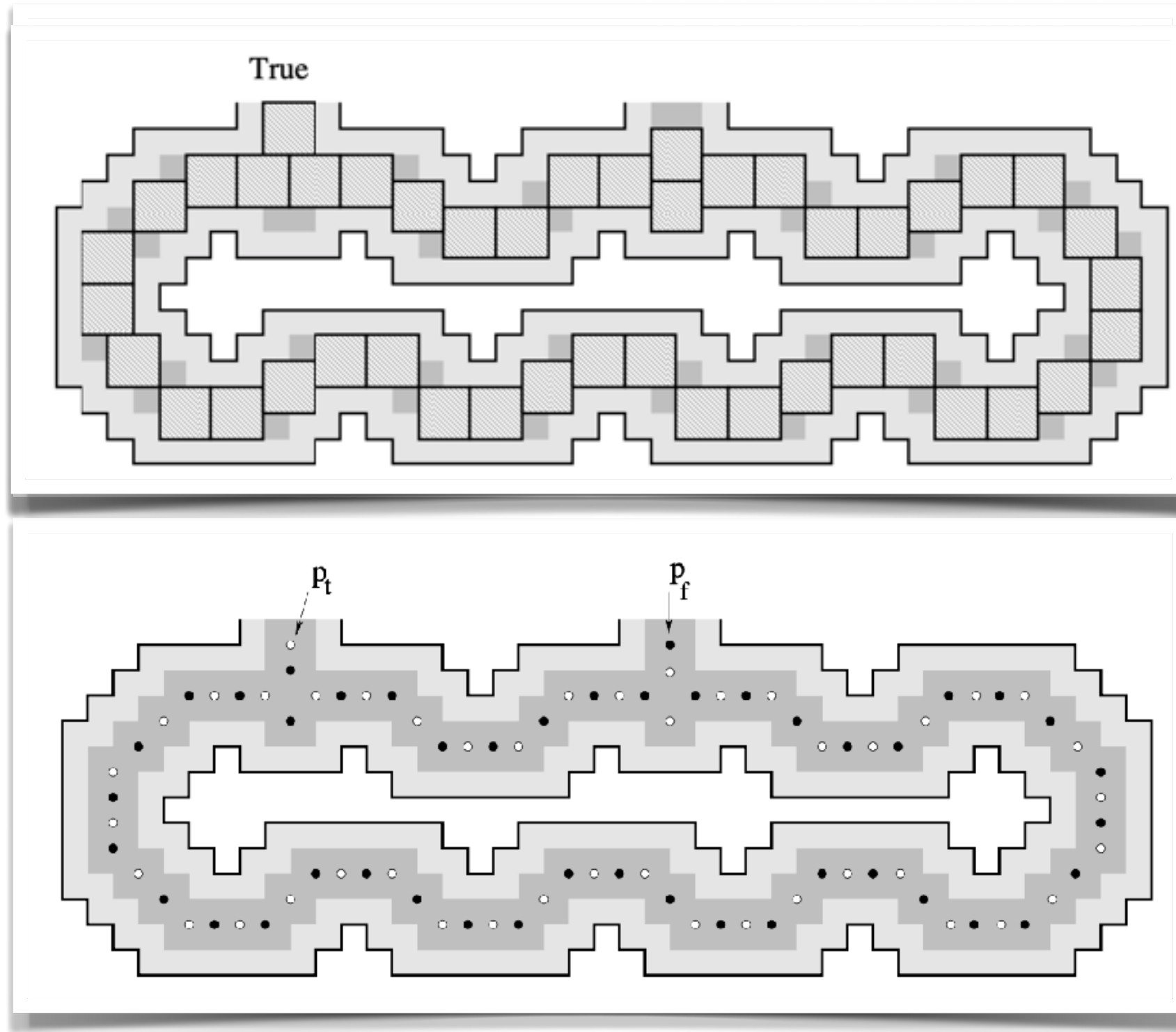
○ Variables (1)



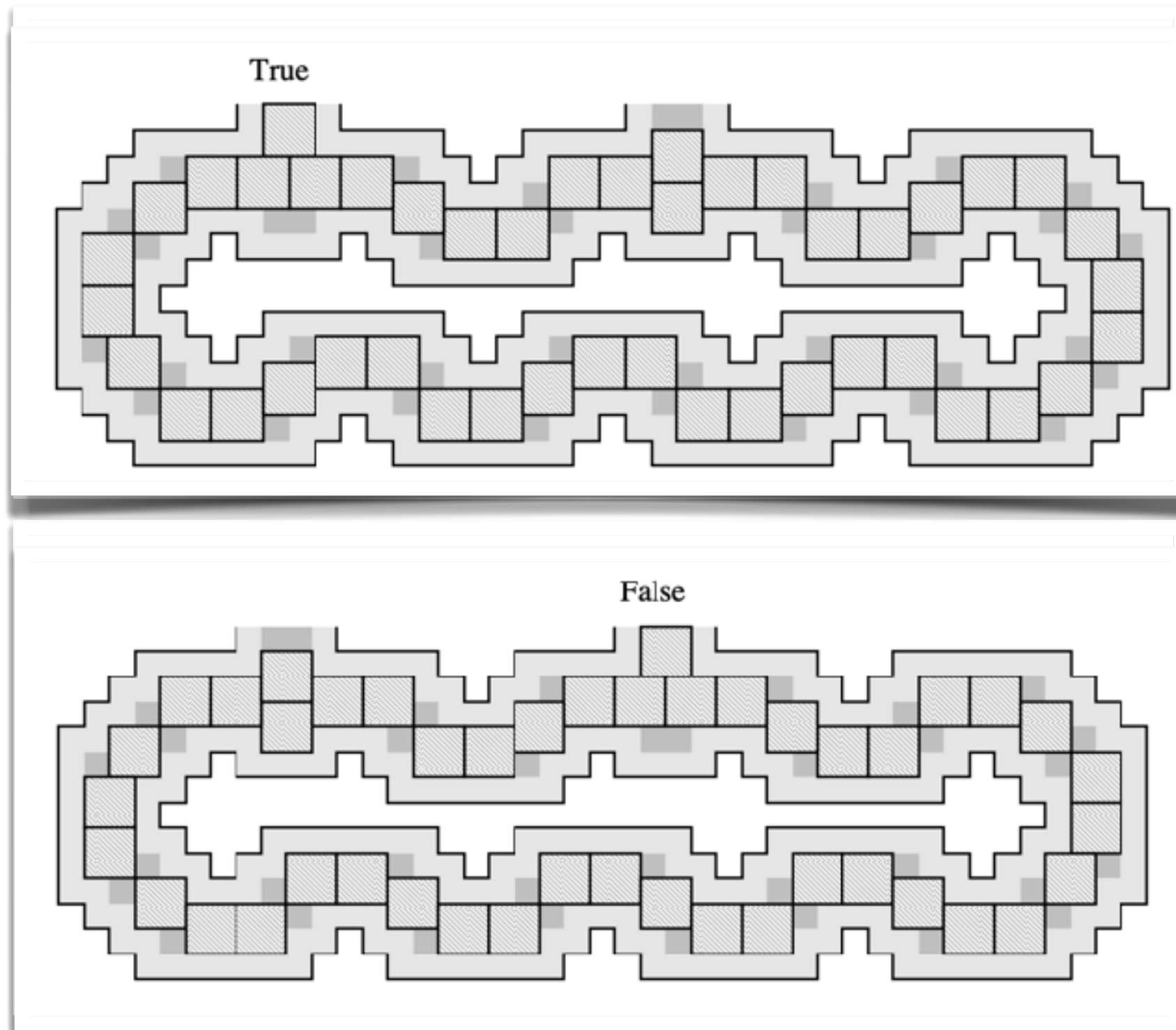
○ Variables (2)



○ Variables (2)



○ Variables (2)



— Connectors

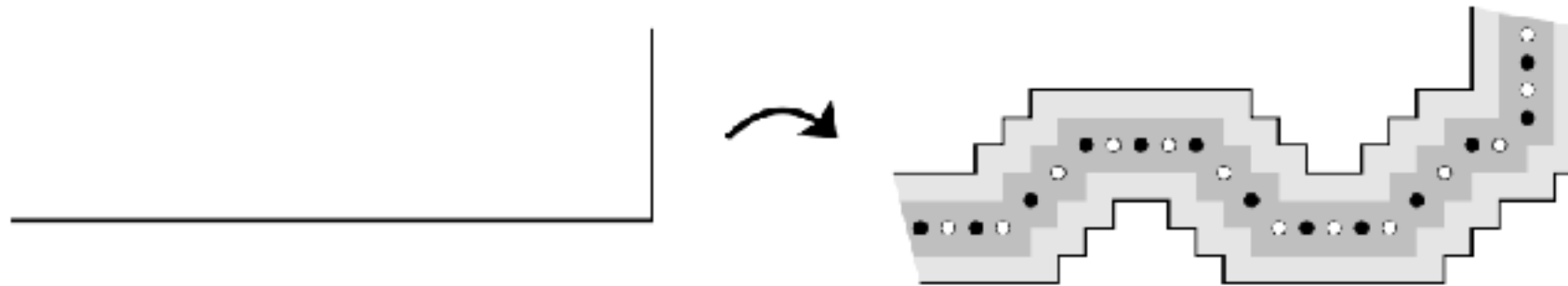
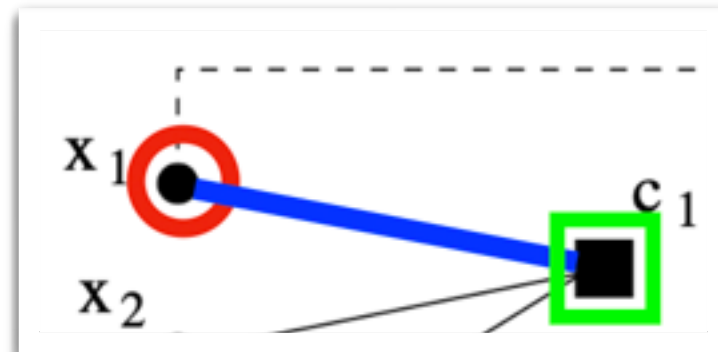


Fig. 3. Part of a connector component (right) for representing an edge between variable nodes and clause nodes (left).

□ Clauses

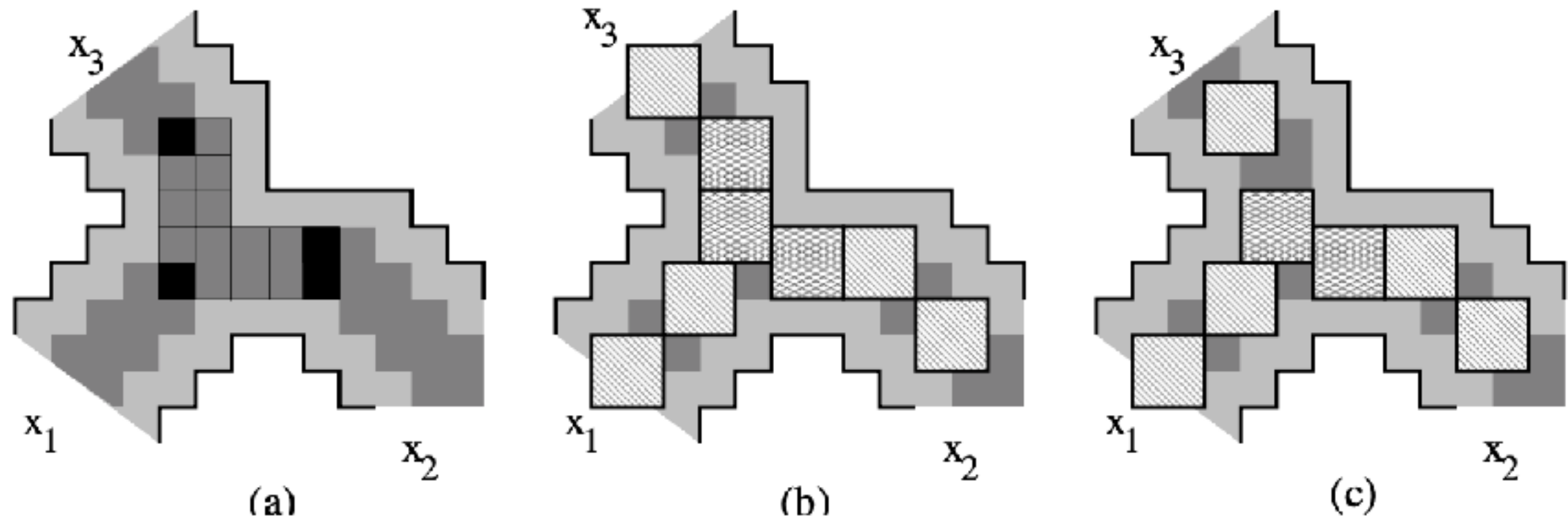
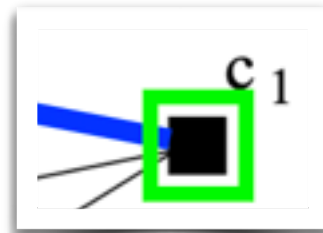
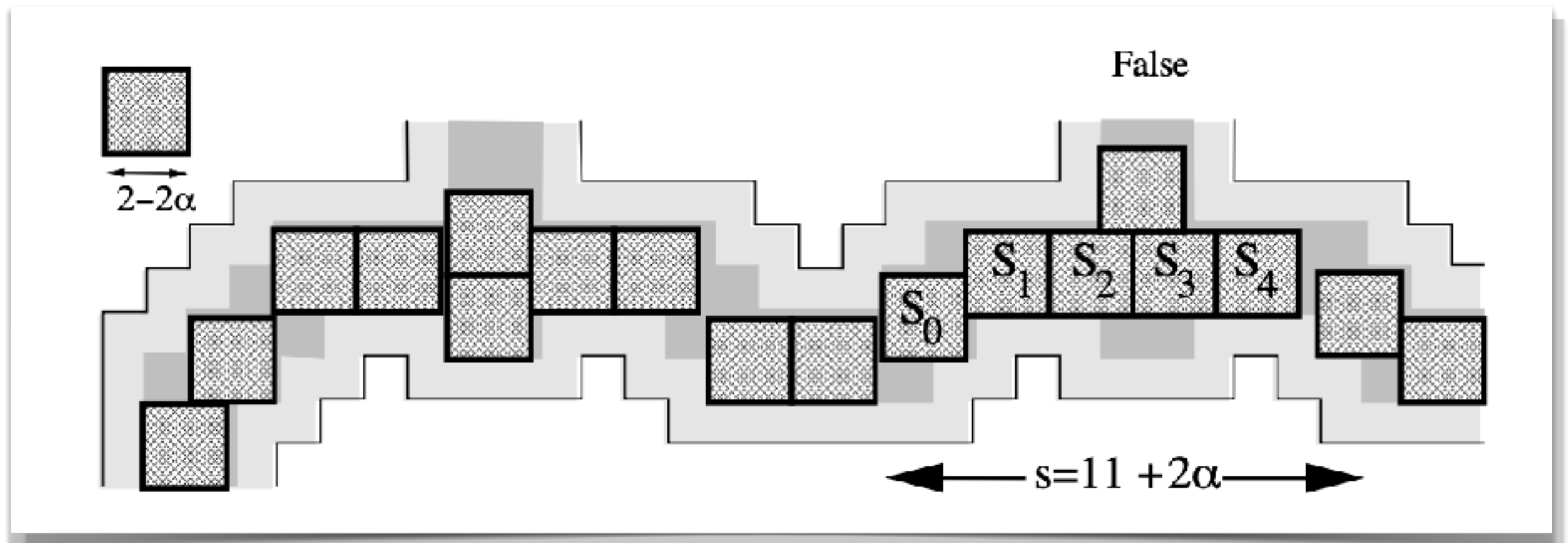


Fig. 4. A clause component for dispersion with boundaries and its receptor region (a); a satisfying placement (b); and an unsatisfying placement (c).

Smaller Squares (1)



Smaller Squares (2)

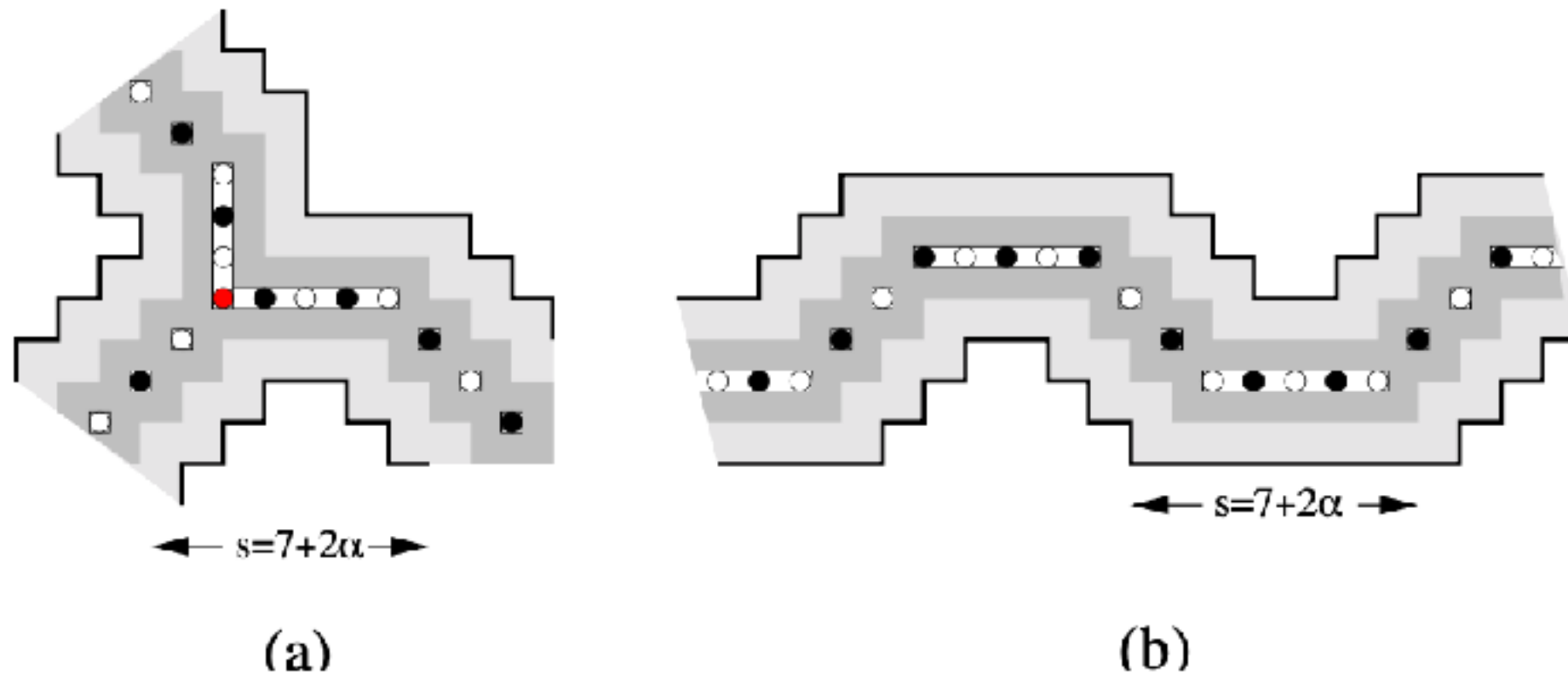


Fig. 7. An upper bound on the approximation factor: clause components (a) and connector components (b).

Other Metrics

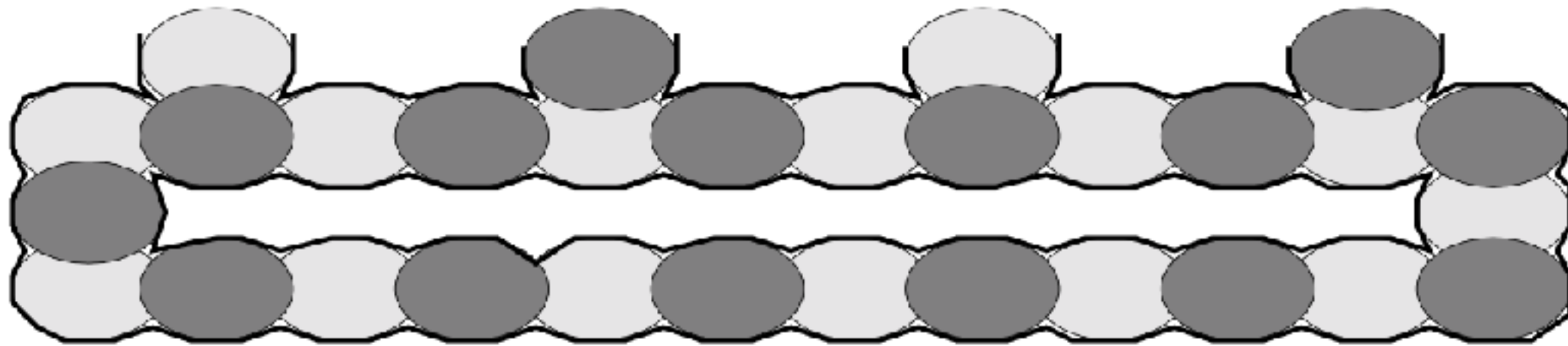


Fig. 9. A variable gadget for the case where the unit ball is an ellipse.

Approximation

THEOREM 9. *For rectilinear geometric dispersion with boundaries of k locations in a rectilinear polygon P with n vertices, Algorithm 8 computes a solution $A_{Dis}(P, k)$, such that*

$$A_{Dis}(P, k) \geq \frac{2}{3}OPT(P, k).$$

The running time is strongly polynomial.

ALGORITHM 8.

Input: rectilinear polygon P , positive integer k .

Output: a set of k locations, such that $A_{Dis}(P, k) := d$ is the minimum L_∞ distance between a location and the boundary, or between two locations.

1. **For all** $(e_i, e_j) \in Par(P)$ **do**
 - (a) Perform binary search for the smallest integer m , $2 \leq m \leq k + 1$, with the following property:
 - For $d_{ijm} := Dist(e_i, e_j)/m$, $AS(P - d_{ijm}/2, d_{ijm}, 6)$ returns a feasible solution for at least k locations at distance d_{ijm} .
 - (b) Let d_{ij} be the distance d_{ijm} for the critical value m .
2. Let d be the maximum d_{ij} for any (e_i, e_j) .

Approximation

THEOREM 9. For rectilinear geometric dispersion with boundaries of k locations in a rectilinear polygon P with n vertices, Algorithm 8 computes a solution $A_{Dis}(P, k)$, such that

$$A_{Dis}(P, k) \geq \frac{2}{3} OPT(P, k).$$

The running time is strongly polynomial.

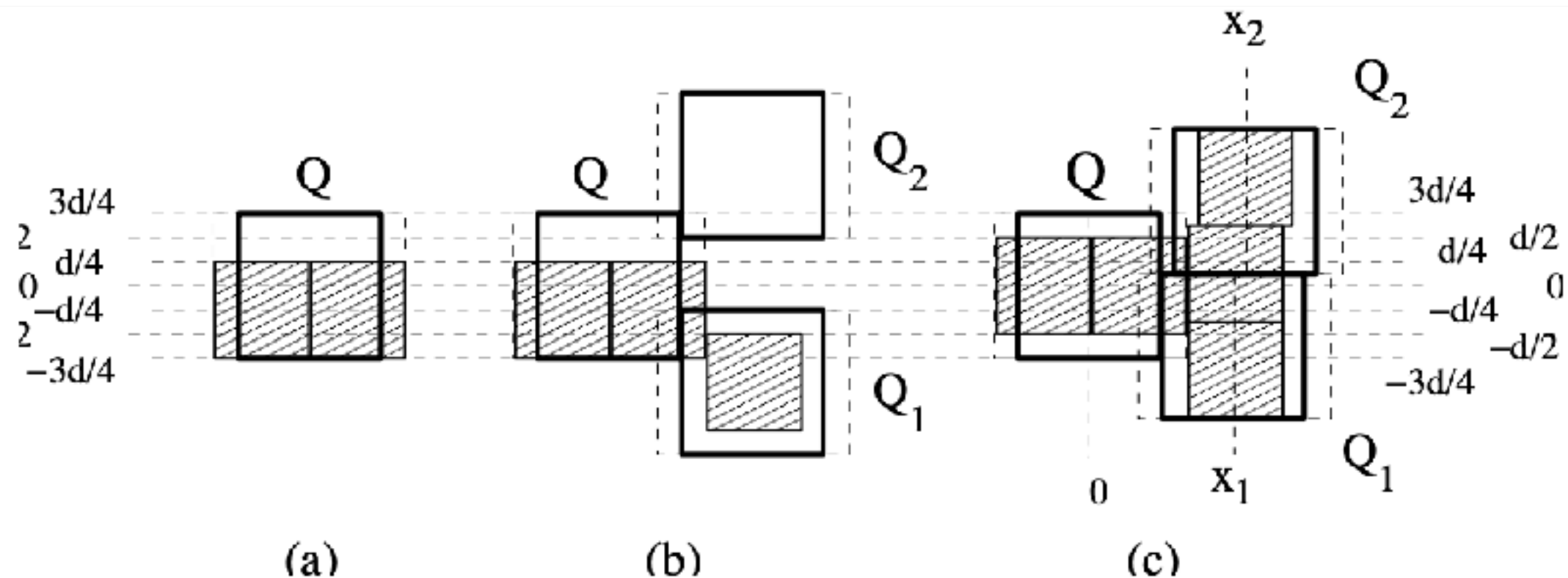


Fig. 12. Constructing a packing of d -squares.

- 1. Introduction**
- 2. Review**
- 3. Extra Packing: Dispersion**
- 4. Extra Tours: Lawn Mowing**
- 5. Relay Placement**
- 6. Coordinated Motion Planning**

Thank you!